NASA-CR-190639

Issues in Designing Transport Layer Multicast Facilities

[illegible]

Bert J. Dempsey
Alfred C. Weaver

p. 33

Digital Technology

August, 1990

(NASA-CR-190639) ISSUES IN DESIGNING TRANSPORT LAYER MULTICAST FACILITIES Interim Report (Research Inst. for Computing and Information Systems) 33 p

N92-31483

Unclas

G3/62 0115107

Cooperative Agreement NCC 9-16
Research Activity No. SE.31

NASA Johnson Space Center
Engineering Directorate
Flight Data Systems Division

ricis

Research Institute for Computing and Information Systems
University of Houston-Clear Lake

INTERIM REPORT

The RICIS Concept

The University of Houston-Clear Lake established the Research Institute for Computing and Information Systems (RICIS) in 1986 to encourage the NASA Johnson Space Center (JSC) and local industry to actively support research in the computing and information sciences. As part of this endeavor, UHCL proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a continuing cooperative agreement with UHCL beginning in May 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The UHCL/RICIS mission is to conduct, coordinate, and disseminate research and professional level education in computing and information systems to serve the needs of the government, industry, community and academia. RICIS combines resources of UHCL and its gateway affiliates to research and develop materials, prototypes and publications on topics of mutual interest to its sponsors and researchers. Within UHCL, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business and Public Administration, Education, Human Sciences and Humanities, and Natural and Applied Sciences. RICIS also collaborates with industry in a companion program. This program is focused on serving the research and advanced development needs of industry.

Moreover, UHCL established relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research. For example, UHCL has entered into a special partnership with Texas A&M University to help oversee RICIS research and education programs, while other research organizations are involved via the "gateway" concept.

A major role of RICIS then is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. RICIS, working jointly with its sponsors, advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research and integrates technical results into the goals of UHCL, NASA/JSC and industry.

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Bert J. Dempsey and Alfred C. Weaver of Digital Technology. Dr. George Collins, Associate Professor of Computer Systems Design, served as RICIS research coordinator.

Funding was provided by the Engineering Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA research coordinator for this activity was Frank W. Miller of the Systems Development Branch, Flight Data Systems Division, Engineering Directorate, NASA/JSC.

Issues in Designing Transport Layer Multicast Facilities

Bert J. Dempsey and Alfred C. Weaver

Department of Computer Science
Thornton Hall
University of Virginia
Charlottesville, Virginia 22903
(804) 924-7605
bjd7p@virginia.edu, weaver@virginia.edu

Abstract

Multicasting denotes a facility in a communications system for providing efficient delivery from a message's source to some well-defined set of locations using a single logical address. While modern network hardware supports multidestination delivery, first generation Transport Layer protocols (e.g. the DoD Transmission Control Protocol (TCP) [15] and ISO TP-4 [41]) did not anticipate the changes over the past decade in underlying network hardware, transmission speeds, and communication patterns that have enabled and driven the interest in *reliable multicast*. Much recent research has focused on integrating the underlying hardware multicast capability with the reliable services of Transport Layer protocols. In this paper we explore the communication issues surrounding the design of such a reliable multicast mechanism. Approaches and solutions from the literature are discussed, and four experimental Transport Layer protocols that incorporate reliable multicast are examined.

[1] The Transport Layer is layer four in the International Standards Organization Open Systems Interconnect (ISO OSI) Reference Model ([28]).

Table Of Contents

1. Introduction

Many distributed applications require efficient, reliable communication between a set of distributed processing entities, or a *process group*. Existing point-to-point protocols force the use of multiple unicast transmissions for group communications. These protocols are ill-suited for multi-party conversations in two fundamental ways. First, they are not designed to take advantage of underlying selective broadcast hardware support available on most modem networks. Second, since failure modes are more complex, the notion of a reliable transfer changes radically under a multi-party communication model, requiring functionality not present in existing point-to-point protocols. In particular, first generation Transport Layer protocols (e.g. the DoD Transmission Control Protocol (TCP) [15] and ISO TP-4 [41]) did not anticipate the changes over the past decade in underlying network hardware, transmission speeds, and communication patterns that have enabled and driven the interest in *reliable multicast*. Much recent research has focused on integrating the underlying hardware multicast capability with the reliable services of peer protocols in the higher layers of the ISO OSI Reference Model. A *reliable multicast facility* is a communication protocol that provides distributed applications with reliable message delivery to a well-defined set of destinations using a single logical address and provides support for *group management.*

Multicasting frames at the Data Link Layer is supported in virtually all standard Media Access Control (MAC) protocols. Local Area Networks (LANs) conforming to the IEEE 802 standards ([25-27]), Ethernet ([21]), and the ANSI Fiber Distributed Data Interface (FDDI) standard ([3]) propagate frames such that all nodes on a frame's originating segment have the opportunity to capture it. Host interfaces to the network support hardware filtering on group addresses, and, as interest in multicast has risen, hardware for efficient address filtering has become increasingly sophisticated. Thus, each Link Layer frame can be delivered, within the constraints of the filtering interface, to exactly the set of destination hosts, or *host group*, for which the frame is intended.

Proposals have been made to extend Network Layer protocols, in particular the Internet Protocol [15], so that host groups can span networks. These efforts to provide a multicast capability for datagram packets traveling over wide-area networks (WANs) focus on host group management and efficient utilization of routing information ([9] [1]). There also exists a body of literature on routing multidestination packets over point-to-point links ([19] [18] [43]).

Considerable complexity arises in translating a machine-level multicasting capability into a reliable multicast facility. At the Transport Layer, the layer traditionally associated with a reliable messaging service, a multicast originator transmits messages using a single network address to a set of endpoints (*contexts*), the *multicast group*. As with unicasting, a Transport

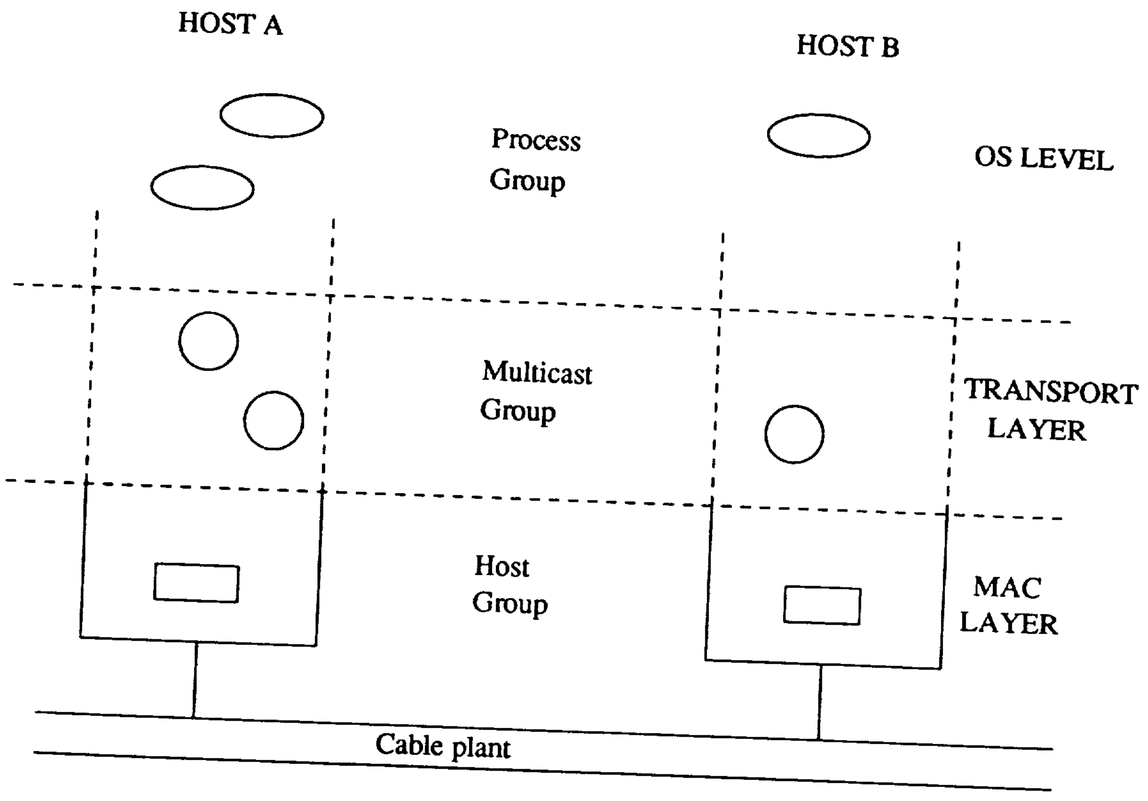

Figure 1 — Multicasting Terminology

Layer multicast consists of two parts: a binding of a message's address to some set of receiver entities and a delivery mechanism to deliver the message to every receiver entity to which its address binds ([9]). The functionality to perform the latter component must come from (next generation) Transport Layer protocols. As for the former, multicast addresses dynamically bind a logical set of communication endpoints to the physical set of endpoints currently listening on the multicast address. This distributed, run-time binding is both powerful and the source of much complexity since some ancillary mechanism outside the data transfer protocol must in the general case manage shifting group membership. This *group management* aspect of the multicast problem has no unicast analogue and requires a management entity that is properly located above (or beside) the Transport Layer. Group management functionality includes handling reliability semantics relating to group membership and other application requirements (e.g., assuring the same message ordering, given multiple senders to the group, at all group members).

1.1. Environments for Multicast

Three different network environments for multicasting may be identified: multicast over a *wide-area network* (WAN), a *multi-segment* environment, and a *single-segment* environment. Multi-segment environment refers to an extended LAN, i.e. one or more LANs connected by Network Layer relay nodes into a single addressing domain. A single-segment environment denotes a network consisting of a single LAN with links, if any, consisting of Data Link Layer bridges.

In a single-segment environment a multicast facility has no routing considerations and can expect (with high probability) nearly simultaneous delivery at the receivers. The high bandwidth and low latency of single-segments allow multicast conversations that are traffic-intensive, such as those in which receivers multicast their control information to the entire group as well as (or including) the transmitter. With the introduction of routing through Network Layer relays, the delivery characteristics experienced by receivers are much less uniform, and consideration must be given to the natural bottleneck at the router in any traffic-intensive scheme. Multicast across WANs introduces the possibility of having point-to-point links in the delivery path, which implies a different routing problem from that for multi-segment environments.

1.2. Multicast Applications

The need for multicasting arises naturally in a number of existing and emerging applications: resource location in a LAN ([2]), distributed databases ([6] [8]), industry process control ([30]), support for distributed operating system services ([29] [10]), replicated procedure calls ([16]), support for real-time command-and-control platforms ([33]), and collaborative development systems ([31]). One taxonomy of multicast applications classifies the behavior of the process groups as either *deterministic* or *nondeterministic* ([32]).

Deterministic process groups require strong data and behavioral consistency between their members. They use peer-to-peer communication, i.e., only members of the group send messages to the group. Examples include parallel processing entities sharing partial results and distribution of status information and coordination among components in automated control programs ([37]). Nondeterministic groups typically do not require the transmitter to be a member of the group. The prevailing model is of a client talking to a functional group of servers. Emphasis is placed on transparent group communication. The client is unaware of new servers coming up or existing members leaving the group. Examples include resource location, replicated procedure calls, and most applications involving group querying and reporting.

One-to-many communication has inherent efficiencies when compared with equivalent service using multiple unicasts. Multicasting allows the source to generate only a single copy of the data, rather than one copy per receiver. Receivers process the distribution concurrently. If connection-oriented service is desired, a single one-to-many connection will most likely be faster and less costly to set up than multiple one-to-one connections. Multicasting thus speeds delivery and saves processing cycles at the source node, bandwidth, and remote host resources.

In shipboard or ground-based command and control environments, for example, signal processing techniques are applied to raw data from sensors and the processed data distributed across high-performance networks to display workstations for human operators ([14]). In [33] a scenario depicting the needs of future Navy platforms, specifically a Tactical Console Display subsystem, is discussed in detail. Twenty display workstations receive multiple data streams, one being a periodic update of the ship's primary track file in which various types of sensor data have been merged. A multicast capability is required to support rapid multidestination distribution of these graphics images, which range from one to ten Megabytes. The real-time constraints present in this environment make multicasting crucial since time does not permit a series of unicasts.

Multicasting offers fundamental benefits besides efficiency. Multicast addressing serves as a run-time binding mechanism for associating a group identifier based on a logical grouping of processes with the actual physical servers. A diskless workstation, for instance, may use, instead of a hard-wired unicast address, a multicast address for the group of boot servers ([9]). The number and location of the servers are unknown at the workstation and possibly change with time. More generally, this de-coupling of logical addresses and physical resources supports distributed data and resources through group querying and reporting.

This functionality will be useful, for example, in achieving substantial increases in network connectivity. A proposal being studied by the National Science Foundation for a National Collaboratory foresees the need for a very rich interconnection between multidisciplinary scientists in order to accelerate the pace and quality of research projects such as mapping the human genome and global change ([44]). In the realm of application development tools, plans are now underway to move up the next step from distributed software development to *collaborative development* in which a number of contractors spread over a wide area will interact daily in the concurrent planning and developing of large software projects. This new software development environment will require multicast in at least three ways. First, there must be rapid file sharing among a number of physically dispersed sites. Second, the substantial increase in the total number of nodes on which project resources will reside will have a dramatic impact on Directory Services. In particular, the need for inquiries to distributed name and route servers will rise. Thirdly, collaborative development will require on-line electronic conferencing and electronic mail distribution lists to which interested parties can subscribe. Both of these applications are most naturally supported by a multicast mechanism. Existing projects such as Grapevine ([7]) and Enchere ([5]) represent first steps toward designing powerful distributed systems that provide the full range of services required for collaborative development.

Reliable one-to-many communication also opens up the possibility of synchronizing distributed processes without incurring the network-wide processing overhead and security problems inherent to broadcasting. If the current work on global time within a network proves successful, this property of a multicast may become especially valuable.

1.3. Reliable Multicast Design Issues

The provision of a general purpose reliable multicast facility involves functionality at several layers of the ISO stack. At the Data Link Layer group addressing must be supported and, in large LANs, routing multicast frames in bridges may be an issue ([19]). At the Network Layer managing host groups and providing for the efficient routing multidestination packets are issues. At the Transport Layer, development of control algorithms, including the efficient collection and coalescing of control information, for one-to-many connections must be addressed. Higher layer protocols are needed to manage the semantics of group membership and other distributed reliability concerns. In this paper we focus on the reliable multicast issues at the Transport Layer and below.

2. Multicast Issues at the Data Link and Network Layers

Several mechanisms are necessary to support the efficient delivery of Transport Layer multicast messages. At the MAC sublayer, group addressing addressing and packet filtering hardware are widely available for sending multidestination frames. Proposals have been made to enhance routing algorithms for both MAC sublayer bridges and Network Layer routers to handle multidestination delivery.

2.1. MAC Sublayer Addressing and Packet Filtering

At the MAC sublayer, multicasting frames requires the capability of binding a frame's destination address to multiple hosts. Standard MAC protocols support this. The 10 Mbit/s Ethernet ([21]) reserves the most significant bit to indicate if this address is a group address and provides the remaining 47 bits to create 2^{47} unique group addresses. Similarly, the IEEE 802 MAC protocols for 802.4 Token Bus ([26]) and 802.5 Token Ring ([27]) as well as the addressing scheme for the ANSI FDDI standard ([3]) support a wide group address space.

In addition to the ability to create MAC sublayer group addresses, however, multicasting MAC frames relies on a host being able to recognize which multicast packets are intended for it. Ideally this *packet filtering* should be done entirely in the network interface hardware since doing it in software is orders of magnitude slower. The advantage of having large group addressing spaces is mitigated by the fact that current network interfaces cannot filter for more than a small number of group addresses, though hardware designers are paying increasing attention to this problem.

Consider, for example, the group address support provided by the TMS380 chipset, a popular token ring interface developed by Texas Instruments for the IBM 4-Mbit token ring architecture ([42]). The adapter board associates three addresses with a node at initialization: the ring station address, a group address, and a functional address. The ring interface also copies any frame with the destination address representing an all-stations broadcast. There are two bit patterns that represent an all-stations broadcast.

The token ring MAC address fields are 48 bits long (Figure 2). The high bit signifies a group address; the next-to-highest bit signifies whether the address is locally administered or

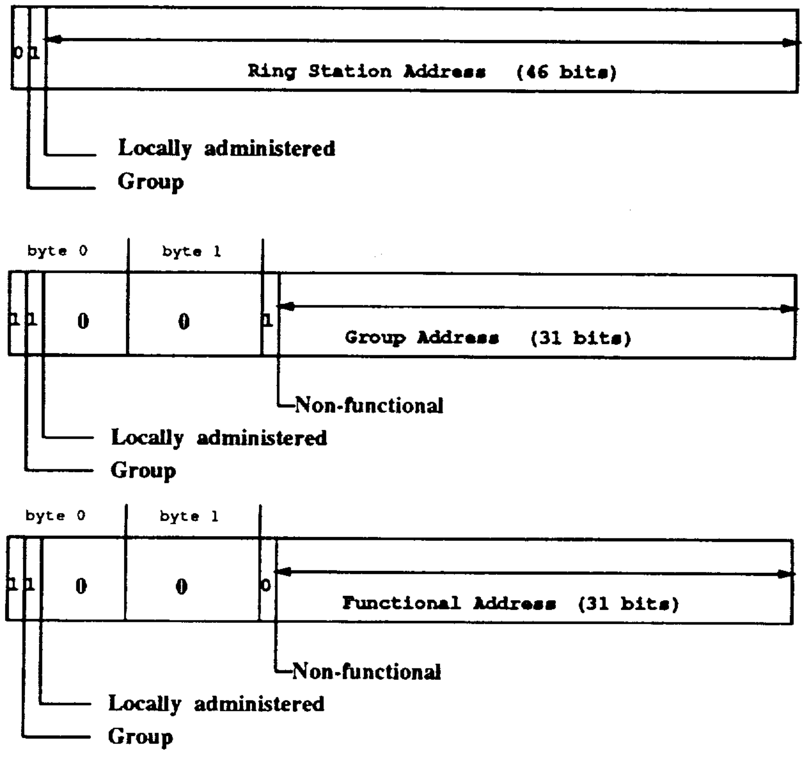

Figure 2 — 802.5 Token Ring Addressing

universally administered. A ring station address uses the lower 46 bits. The group address format has a fixed bit pattern in the upper 17 bits and the group address in the lower 31 bits. Functional addresses also have a fixed bit pattern in the upper 17 bits and 31 bits of system-supplied address. When the destination address is a functional address, the station matches its functional addressing mask against the functional address. If any bit position is set in both, the station copies the frame from the network. Thus, functional addresses are encoded in a bit-significant manner, and any station may filter for any or all of the 31 functional groups. Five functional groups have already been designated by the IBM token ring architecture for special purposes (e.g. use by a bridge, network manager, or active monitor). A multicast facility has approximately 27 distinct group addressing filters — the 26 unclaimed functional addresses and the one group address— supported in hardware at each node at any one time.

2.2. Routing

A LAN is constrained by limits on the number of stations, maximum distance between any pair of stations, and maximum traffic loads. Thus multiple LAN segments are often needed for a single community of network users. These multiple LANs are connected by intermediaries known in the ISO terminology as *relays*, and a relay may be present at any layer of the ISO OSI Reference Model. If the relay shares a common layer *n* protocol with other systems, but does not participate in a layer $n+1$ protocol in the process of relaying information, it is known in the ISO terminology as a *layer n relay*. Common terminology denotes a Physical Layer relay as a *repeater*, a Data Link Layer relay as a *bridge*, a Network Layer relay as a *router*, and any higher layer relay as a *gateway*. While this terminology is common, it is not used universally and one should be aware that the term *gateway* is sometimes used in the literature to describe a relay at any layer ([35]).

2.2.1. Network Layer

Routing techniques for multicast in store-and-forward networks have been examined in a number of contexts. Historically, these multicast routing techniques were first examined for point-to-point networks ([43]). Most strategies are built around spanning trees, a natural solution to the problem of taking an arbitrary topology and producing an edge set in which there exists exactly one path between any pair of nodes, i.e., eliminating cycles. In his proposal for an Internet multicast, Deering ([19]) has put forth extensions to the two standard routing algorithms for Network Layer routers—distance-vector routing and link-state routing— using in the former case refinements to *reverse path forwarding* ([18]).

Networks in which, instead of point-to-point, multiple-access links connect routers, or *bus-based networks*, have different characteristics and routing criteria than point-to-point WANs. Cost is an important consideration in general for WANs, but plays no role in routing through a bus-based network, where packets do not incur tariffs. An excellent summary of techniques for multidestination routing in this environment appears in [23]. Recent work on this problem also appears in [34], though the authors point out that their algorithms for *multicast trees* are more appropriate for multiprocessors and multichannel LANs than for interconnected LANs or MANs since the algorithms depend on maintaining complete knowledge of the network topology at all network nodes.

LAN-based multicast has generally been explored with the assumption that no Network Layer routers are present in the delivery path. Extension of these techniques to the simple case of a multi-segment LAN in which there exists a maximum of a very few routers (perhaps two) between any destination node and its source may be fruitful, but remains an open research question. Multicast techniques and implementations must mature before more work is done on routing issues within multi-segment environments.

2.2.2. Data Link Layer

The IEEE 802 protocols specify two sublayers within the ISO Data Link Layer, the Medium Access Control sublayer, located next to the Physical Layer, and the Logical Link Control (LLC) sublayer above. As intended by the 802 Committee, the term *bridge* refers not just to a Data Link Layer relay, but more specifically to a relay operating below the MAC sublayer service boundary within the Data Link Layer. This definition ensures that the relay will operate independently of all LLC and higher layer protocols. Bridges are store-and-forward routing switches that attach to two or more electrically independent cabling LAN segments. Hence a frame arrives on one of the cable segments, the *incoming link*, and is forwarded onto one or more *outgoing links*. A *bridged LAN* refers to a LAN in which all relays are bridges (or repeaters).

Two routing algorithms for bridged LANs have been endorsed by the IEEE 802 Standards Committee: the IEEE 802.1 Transparent Spanning Tree (TST) Scheme and the IEEE 802.5 Source Routing Scheme. *Transparent bridges*, bridges as defined by the IEEE 802.1 Medium Access Control Bridge Standard ([24]), provide transparency in the sense that end nodes do not participate in routing decisions. Instead TST bridges use a distributed algorithm to transform the arbitrary mesh topology of the given network into a single, acyclic spanning tree through which frames are forwarded. These bridges maintain a forwarding database of the location of nodes as determined through examination of the source addresses in frames. Topology changes are detected by intra-bridge communication, and a new spanning tree determined. The bridges self-configure upon initialization and even recover if misconfigured by human installers.

Source routing is based on including the route to the destination node(s) in a variable length field of the frame. Under this scheme, a bridge performs string matching on the routing field to determine to what links, if any, this bridge should forward the frame. Source routing has a number of advantages. Perhaps most important of all, source routing bridges are relatively unaffected as the size of the network grows and transmission speeds increase unlike TST bridges, which are tied to address-table maintenance and look-up. A major limitation to source routing is that a key element of dynamic route discovery by the source host consists of broadcasting frames throughout the network to explore all possible paths tò the target ([22]). Consequently, its use is more appropriate in bridged LANs of small diameter.

In a bridged LAN of small diameter multicast packets are simply broadcast to all segments. The abundance of switching resources and bandwidth compensate for the inefficiency of delivering packets to segments where no receivers exist. Filtering hardware offloads the host in the task of discarding packets. Thus, added complexity in the bridge routing algorithms to achieve *scope-controlled* multicasting, a multicast that propagates a fixed "distance" from the originator instead of throughout the network, cannot be justified for small diameter networks. It follows that attention should be focused on the TST Scheme and not the Source Routing Scheme.

In [19] the authors propose extensions to the TST Routing Scheme to accommodate efficient multicast for large bridged LANs (on the order of 10 segments). The scheme augments routing tables to handle multicast addresses and dictates that the members of the multicast host group, G, issue periodic *membership report* packets by which bridges learn the links on which to forward packets with destination G. In this way bridges learn the paths for multicast packets and confine multicasts to portions of the network where members of the destination group reside. The overhead of sending membership reports in order that bridges can learn about the location of group members is shown to be very manageable. The primary drawback to this proposal is the loss of transparency in the hosts. The authors argue that the appropriate functionality may appear in future LAN interfaces and can in any case be provided by modifications to LAN device drivers, but for current systems such modification may not be deemed justifiable.

In [40] the authors develop algorithms that allow the use of (non-standard) bridges in extended LANs of arbitrary topology without confining the traffic to a single spanning tree. The scheme depends on decomposing the network graph into some number of spanning trees, numbering them, and then marking each packet as traveling on a single tree. The TST-bridge technique of building routing tables based on the source address of packets passing the bridge ([4]) can be preserved while traffic flows along multiple paths. Given the ability to perform such a multitree decomposition of the network, the authors go on to present a routing algorithm for efficient (*scope-controlled*) multicasting. The algorithm does depend on two-way communication to resolve a path so that hosts involved in the exchange must transmit at some guaranteed minimum rate in order for the bridges to retain the proper routing information.

The idea of using multiple spanning trees has a number of appealing characteristics. Like source routing, it allows dynamic load balancing, leading to better overall network performance, and in the case of a link failure, it enables a connection to switch very quickly to another route. Unlike source routing, all the preparation cost of determining and numbering a set of spanning trees (i.e. a set of well-known routes cached at each node) can be confined to network initialization time. Addressed-based table look-up suffers the same drawbacks as the TST Routing Scheme, only the address tables are even larger with multiple tree forwarding. And, of course, an implementation of this scheme would require special purpose hardware bridges, which may be expensive and risks interoperability problems with existing networks based on international standards.

3. Transport Layer

Traditional Transport Layer (unicast) service shields higher layers from the details of the underlying unreliable network, including transparent recovery from lost or duplicated data.

Since packets can be lost, the receiving context in most point-to-point Transport Layer protocols sends control packets back to the sender. These packets typically include an acknowledgement of received data (error control) and an indication of the availability of buffers for more data (flow control). In order to recover from lost control packets, the sender usually employs a timer. If the timer expires before the arrival of an expected control packet, the control packet is assumed lost, and the sender takes actions accordingly, e.g. requesting the receiver to issue another control packet.

The presence of multiple receiving contexts — reliable Transport Layer multicast — complicates this scenario. First and foremost there are group membership questions. A unicast address binds to a single, unique endpoint within the network. If that endpoint does not exist at connection set-up or fails during a data transfer, then the transmitter easily detects the failure since no control information arrives. With the dynamic binding of multicast addresses, a partial connection is possible. Higher level mechanisms must ensure that, in any given exchange, group membership is 'correct'. Even if membership is 'correct', a Transport Layer transmitter may not have a separate control channel for each receiver and hence can not know when all multicast group members have reported their status.

The technique of making the control channel reliable by timing out lost control packets encounters problems when extended to the multiple receiver case. First, the timer must be based on the maximum of a set (possibly of unknown cardinality) of roundtrip times. Second, if a time-out occurs and a control packet from each receiver has not been received, then the protocol may act on the partial report from the receiver group and risk making a wrong decision that degrades the efficiency of the transfer or, worse, loses data due to the premature release of a transmit buffer. Alternatively, the receivers not responding can be offered another chance to respond. This approach, however, leads to the problem of how to contact these silent receivers. Two possibilities exist: (1) the sender initiates a new response from all receivers, which may be expensive in terms of network resources and may well result in another partial report from the set; or, (2) the sender attempts the potentially prohibitively slow action of unicasting requests for control packets to each silent receiver, assuming that all receivers are known individually. A reliable Transport Layer multicast mechanism must specify one-to-many (flow, rate, and error) control algorithms that are robust and efficient in the face of partial updates.

For any many-to-one data flow within a LAN (e.g. collecting acknowledgements from the receiver set), the phenomenon of *network implosion* must be addressed. Under any transmitter-driven control scheme the set of multicast receivers will tend to synchronize the sending of their control packets. Synchronized transmission can result in bursts of traffic on the network and the inability of the multicast source's network interface to capture frames arriving back-to-back.

Even if all receivers send control information, the multicast transmitter must collate the multiple status reports into directives that drive the multicast transfer. When the sender determines that data has indeed been lost in transit to some subset of the receivers, for example, the data must be retransmitted. If retransmissions are multicast, when a single receiver or a small number of receivers causes retransmission of a data packet, there is much work lost in resending data to the receivers who have already successfully received it. If retransmissions are unicast, the sender may have to frame and send a large number of copies of the same data.

3.1. Four Reliable Transport Layer Multicast Mechanisms

Reliable Transport Layer multicast mechanisms must first ensure effective collection of control information from multiple receivers and secondly specify robust one-to-many control algorithms. Below we examine four Transport Layer protocols that address reliable multicasting. These protocols emerge from different design philosophies and assumptions about use, performance, and environment. Perhaps the most important difference to note in comparing their approaches to reliable multicast are the assumptions about group management support from higher layer protocols. Two of the four assume that the transmitting context has been supplied with an explicit list of the receivers at the beginning of the data transfer; the other two do not posit any group management entity.

3.1.1. CP

The Transport Layer reliable multicast protocol proposed in [17], called here CP, represents a straightforward, but detailed attempt to handle reliable multicasting by having the multicast sender manage separate transmit windows for each receiver. The protocol assumes that process group membership is managed by some mechanism that allows the Transport Layer user to state the group membership and lock onto it for the duration of each individual exchange. The transmitting multicast context therefore has a list of group members. CP supports a range of reliability requirements, gives explicit consideration in its design to the possibility of internet links of low-bandwidth and/or a point-to-point nature in the delivery path, and has two proposed service interfaces.

3.1.2. Versatile Message Transaction Protocol

The Versatile Message Transaction Protocol (VMTP) ([12]) is designed as a next generation protocol to accommodate communication strongly oriented toward request-response behavior and uses the transaction paradigm as the basis of all communication. Reliable multicast transactions are defined as transactions with group entities in which at least one response from the multicast group is received. Responses after the first one are buffered for the user and delivered if requested. Hence, messaging service reliability depends, beyond the initial response, on the reliability of user-level transactions. The V Distributed Operating System ([10]), to which the development of VMTP has been closely coupled, defines a service interface that includes process group management primitives. VMTP itself has an integrated management facility that handles creating, modifying, and querying for group entities (multicast group identifiers).

3.1.3. Xpress Transfer Protocol

The Xpress Transfer Protocol ([13][36]) (XTP) is a lightweight transfer layer (the transfer layer being defined as the Transport and Network Layers merged) protocol being developed by a group of researchers and developers coordinated by Protocol Engines, Inc. It is designed to provide the end-to-end data transmission rates demanded in high speed networks such as FDDI and the gigabit/sec wide area networks without compromising reliability and functionality, including in particular, support for reliable multicast. XTP intends to accomplish its goals through streamlining the protocol, combining the Transport and Network layers, and utilizing the increased speed and parallelization possible with a VLSI implementation ([39]). XTP defines a reliable multicast mechanism such that a transmitting context, knowing only the group address, can perform a flow, rate, and error-controlled one-to-many message delivery. Like VMTP and unlike CP, the mechanism described has been carefully designed so that reliable multicast imposes a minimum of overhead on unicast protocol processing. The reliability guarantee is fragile in the sense that transmit buffers are released based on estimations of the maximum roundtrip time between the sender and the receiver set.

3.1.4. NAPP

A mechanism based on Negative Acknowledgement with Periodic Polling (NAPP) ([38]) takes the novel approach of having background daemons at each receiver that assure progress and periodically send liveness messages to the source during a multicast distribution. The mechanism assumes that a one-to-many 'virtual circuit like connection' has already been established; thus, the sending context has explicit knowledge of the receiver set. Receivers multicast control information so that all group members overhear each other, and each control message that reaches the multicast source contains a report on all receivers' sliding windows.

This relaying of control information reflects a fundamental assumption underlying the design of NAPP, namely that the failure of one receiver to receive a packet strongly suggests that others have missed the packet as well.

3.2. Flow Control

Flow control refers to the receiver's ability to throttle the source in order not to overrun the available buffer space on the receiving host. A multicast exchange's flow control must be governed by the minimum of the flow control parameters for all the receivers in the exchange. The alternative is to allow a situation in which some subset of receivers is deliberately overrun, a strategy that would normally be counterproductive.

Maintaining proper flow control parameters at the sending context is particularly important since hardware improvements have produced networks with vanishingly low bit-error rates, meaning the majority of errors on these networks will occur due to incorrect flow control. The control information collection strategy should ensure that the transmitter knows about or quickly learns the correct flow control parameters at connection set-up. If for any reason during a data exchange (i.e. early release of transmit buffers) the receiving group is pruned, the transmitting context should ideally recompute the new minimum flow control parameters since some slow receivers may have been dropped.

XTP, CP, VMTP, and NAPP base flow control on the most limited receiver in the receiving group. The success of their flow control algorithm therefore depends on the effectiveness of their control packet collection schemes. Unlike CP and NAPP, XTP and VMTP do not assume that the multicast transmitting context has explicit knowledge of the receiving group and therefore cannot know with absolute certainty whether all receivers have reported their flow control parameters or not.

3.3. Collection of Control Packets

3.3.1. Simple Reporting: CP

The CP protocol takes the simplest approach to structuring the flow of control information by having multicast receivers unicast their control packets to the source, which knows the receiving set and manages a control channel for each receiver. The strong reliability guarantees possible under this scheme result from (1) the maintenance of a control channel for each receiver and (2) the underlying assumption of a powerful group management support facility. This facility maintains a lock on the multicast group membership for the duration of an exchange and notifies the transmitter should a server leave the group abnormally. As for network implosion, the designers of CP acknowledge the problem, present some mathematical analysis of it, suggest some general approaches to dealing with it, and finally leave it to implementors of the protocol to solve.

3.3.2. User-Level Responses: VMTP

In the VMTP unicast, a transaction starts with a client issuing a request to a server entity. At the server, on-demand connection set-up creates a transaction record upon receipt of the request. It is expected that a response packet containing the user-level response data will usually be quickly generated at the server, and that this response packet will function as an acknowledgement to its associated request. Otherwise, based on a time-out, the client sends a demand for an explicit acknowledgement to the server, who responds immediately. In this way, the client is assured that the delay is due to server processing and not because the request was lost.

VMTP's multicast capability focuses on compatibility with unicast mechanisms. A multicast transaction follows the same sequence of events as described above for unicasts except that the client sends its request to a group entity. The VMTP sender sets its timer upon issuing a request. Receipt of the first response disables the timer, and thereafter the VMTP client awaits responses without taking any further action. If the timer expires before any response arrives, the VMTP client issues a demand for an immediate acknowledgement from the group of servers. No special mechanisms address network implosion, though user-level acknowledgements generally produce a much greater variance than control packets generated within the communications protocol itself, making implosion less likely.

VMTP's 1-reliable multicast primitive is tailored toward the protocol's target environment of rapid exchange of small amounts of data over a network with low error rates (e.g. remote procedure calls over LANs). (The protocol accommodates large requests and responses through *packet groups.*) For the common case of a single packet multicast request, the VMTP multicast provides a low-overhead service. The application-level transaction determines reliability beyond the initial response, which indicates a high probability that the members of the server group will see the request. For the *k*-reliable semantics of multicast introduced in the V System, this 1-reliable primitive appears to be adequate. The specification of VMTP [11] does not explicitly describe how to provide flow-, rate-, and error-controlled one-to-many delivery of multi-packet requests, though it could be argued that the existing protocol features are adequate to build such a service.

3.3.3. Damping: XTP

The XTP multicast control scheme dictates that multicast receivers generate control (in XTP parlance, CNTL) packets upon detecting corrupted or out-of-sequence data packets. The transmitter may also set request bits in out-going data packets to force each receiver to issue a control packet. At the transmitter, in-coming CNTL packets are coalesced by recording the minimum of rate and flow parameters and the minimum of the byte-based sequence number (*rseq*) that indicates the highest consecutive sequence number received without error at the issuing receiver. A timer using estimations of the maximum roundtrip time between the sender and the receiver set determines the intervals between processing the cumulative control information. The algorithm controlling these intervals represents a crucial element in any implementation of XTP multicast. Unless transmit buffers are mistakenly released early, the XTP error control algorithm ensures that all correctly functioning receivers will eventually receive the multicast distribution.

In XTP a multicast receiver issues a control packet to the group address. Other receivers, as well as the transmitter, see the control packet and dequeue any control packets that they have which contain an *rseq* value greater than or equal to the overheard value. This process, called *damping*, lessens the number of superfluous control packets flowing to the transmitter and addresses network implosion for the case where a large set of receivers drop the same data packet (e.g. a packet that was corrupted when transmitted). Damping does not presuppose any knowledge of group membership at either the sender or the receivers.

As defined in Revision 3.4, XTP's multicast strategy is applicable only to the single-segment LAN environment as, for one thing, damping may cause unnecessary congestion at a router when multicast members sit on both sides of the router. Since retransmission follows a go-back-N policy, XTP multicast would be inappropriate for networks with low bandwidth or high bit-error rates. On a noisy channel an XTP multicast may find it difficult or even impossible to make forward progress ([36]).

The primary disadvantage to the proposed damping mechanism results from its fragility due to timing considerations that may vary widely over disparate environments. It is not clear that a node, R, can receive CNTL packets and perform quickly enough the processing necessary to locate and dequeue R's own CNTL packet. These timing concerns may seriously jeopardize the robustness of the multicast mechanism. *Slotted damping*, the implementation technique of introducing a random back-off time before a receiver generates its CNTL packet, may be an improvement, but it suffers from the same environment-specific timing dependencies. Slotted damping forces receivers to have some estimate of the size of the group. Otherwise unnecessary delays result in receivers for small groups, or secondary collisions can be expected in large groups. In short, damping remains an unproven technique.

If experience reveals damping to be unsuitable, the designers of XTP will consider alternatives. One approach is the collection of control information through relaying information back to the transmitter via a control channel structure such as a tree or a ring. The multicast sending context, the receiving contexts acting independently, or group management could establish and maintain the relay route. If the structure is created at connection set-up time, long-lived connections are preferable since they amortize more efficiently than short-lived connections the cost of set-up.

Consider the following scenario. Each group will contain a small number, say four, of special receivers, called *collectors*, from which any sender to the group will receive all control packets for the transfer. When a processing entity joins a group, it is given a collector's address, to which the new group member will unicast its control packets. Collectors coalesce control packets and relay (unicast) them in a single composite control packet to the multicast sender. A collector needs only enough knowledge to set up its address filters correctly and some logic with which to coalesce control packets. From the multicast sender's viewpoint, the data transfer is considerably simplified. The sender establishes a connection with the group and receives a fixed number (here four) of control packets on each sender-generated request for control packets as well as error reports whenever errors occur ([20]).

This caching strategy offers many benefits. Four collectors would widen the bottleneck at the source host's network interface by a factor of four. The two-step unicasting of control packets avoids the potential for generating a large number of packet interrupts at participating hosts. This consequence is inherent in the schemes that have receivers multicast their control packets to the group (like XTP). Collectors perform a sort of localized suppression of control packets in coalescing packets. This feature would be useful in reducing traffic through routers for a group residing on multiple segments and in partitioning the problem of collecting control packets for large groups on a single-segment LAN. Moreover, the collector algorithm could possibly be designed to collapse to a simple scheme of receivers unicasting responses directly to the sender for small groups that do not need two-step control packet reporting. The delay of relaying packets, especially on single-segment LANs, the heavy processing duties of the collectors, and the overhead of managing the relay structure represent the primary drawbacks to this idea.

3.3.4. Polling: NAPP

NAPP and CP have similar design goals in the following sense. Both protocols emphasize a high degree of reliability in multicast data delivery at the expense of producing lightweight, fast protocols. (For XTP and VMTP, the trade-off is roughly the opposite.) In NAPP the multicast source transmits data and performs retransmissions based on control information from the receivers. Like XTP, NAPP uses multicast control packets so that receivers may monitor each other's state and thereby reduce the amount of control traffic. NAPP receivers, however, interact in a far more complex manner than the simple damping behavior found in XTP.

Receivers issue three packet types for control information: ACK, PACK, and SREJ. All three are multicast so that receivers overhear and monitor each other's state upon transmission of every control packet. All three contain a *state vector* reporting the highest in-sequence packet received at each of the receivers. The data source uses the in-flow of state vectors to decide when to slide forward its transmit window.

Some terminology is needed for the discussion of NAPP that follows. Let M be the maximum number of packets that can be outstanding, that is, pending to be acknowledged at any one time. Let V_i be the first in-sequence packet not received at receiver i. Finally, let W_i be the window of receiver i consisting of packets sequenced V_i, ..., $V_i + M - 1$. All timers are assumed to have a granularity of milliseconds.

A receiver issues a poll-cum-acknowledgement (PACK) every T_{pack} milliseconds. The PACK is numbered V_i (expressed here as PACK(V_i)) and serves to solicit (re)transmissions, if any, of packets in W_i and acknowledges the packets in the range $V_i - M$, ..., $V_i - 1$. A PACK is rescheduled for T_{pack} milliseconds later upon reception of a packet in W_i, upon transmission or receipt of an SREJ(m), m W_i, or upon receipt of a PACK(q), $q \geq V_i$. Thus, PACKs serve as sort of background daemons that are never actually transmitted as long as data continues to flow to the receiver.

An SREJ(m) packet is scheduled for transmission by a receiver as soon as message m is detected as being lost. However, any SREJ packet is transmitted at its scheduled transmission time only with probability P and otherwise rescheduled for T_{srej} milliseconds later. When a receiver, R, overhears another receiver's SREJ(m), if message m is known to be lost already, then its own SREJ(m) is rescheduled for some time later, presumably putting off SREJ(m) long enough that the overheard SREJ(m) will have gotten message m retransmitted in the meantime. Any scheduled SREJ(m) is dequeued upon reception of m. If message m has already been received at R, the overheard SREJ(m) is ignored. Otherwise, message m is now perceived to be lost, and a SREJ(m) is scheduled for later transmission. Furthermore, receiver R checks to see if any messages from V_i, ..., m - 1 are lost. Thus, the reception of SREJ(m) at the source serves to acknowledge (possibly redundantly) $V_i - M$, ..., $V_i - 1$.

The third component in the trio of control packet types is an ACK(p) packet. ACKs are positive acknowledgements that receivers issue upon receiving some number of packets in sequence. To ensure reliable delivery, upon transmitting an ACK, a receiver reschedules the transmission of the same ACK for T_{ack} milliseconds later. ACKs are not necessary for the correct working of the protocol, but they do speed up the process of conveying acknowledgement status and advancing the source's transmit window. ACK(V_i) acknowledges $V_i - M, \ldots, V_i - 1$ at the source. Also, receivers overhear other receivers' ACKs and use them to monitor status in ways similar to those outlined for PACKs and SREJs.

Though there are more aspects to NAPP, this description gives the flavor and the most important aspects of its operation. In an actual implementation of NAPP, much attention must be given to the mechanisms to determine the correct settings for its many timers; the paper describing NAPP ([38]) does not focus on these implementation details, but instead notes the relative lengths of timers, e.g., $T_{\text{pack}} > T_{\text{ack}}$. A primary drawback to NAPP's approach is the management of adaptive timers in the face of dynamic system parameters, changes in group size, and connections made by multicast sources of varying processing power. The defaults for the timers that drive the background daemons at each multicast group member may be inappropriate for a particular connection. Short transfers will suffer unpredictable delays and/or periods of temporary instability as timers adapt.

3.4. Error Control

Error recovery at the multicast source must address whether to unicast or multicast retransmissions and whether to selectively retransmit or use go-back-N. Go-back-N requires less processing to determine the exact data that was lost, but risks generating large amounts of data if multiple requests for the same data are processed at the multicast source. Multicasting retransmissions burdens up-to-date listeners with processing duplicate packets. Unicasting to the unsuccessful receivers forces the sender to frame and send multiple copies of the data and to know how to address individual group members.

XTP uses go-back-N multicast retransmission and relies on an implementation having a robust method for calculating of the proper processing checkpoints. VMTP does not specify multicast retransmission policy, though the logical choice seems to be selective multicast retransmission. NAPP provides for selective multicast retransmission. Multicasting the data follows from the NAPP designers' belief that the loss of data at one receiver strongly suggests loss of data by other set members. Selective retransmission makes sense given the sophisticated interaction between NAPP receivers, which reduces the possibility of retransmission requests overlapping. Finally, CP employs a more elaborate mechanism. In deference to the possibility of a multicast exchange over low bandwidth delivery paths, retransmissions are unicast to individual receivers unless the proportion of failed deliveries to group size is larger than some user-supplied threshold value. In the latter case, retransmissions are multicast to the entire group. The threshold value would be set based on the number and relative dispersion of the host group, lower for groups on single-segment LANs and higher for groups on extended LANs or across internetworks.

References

1. L. Aguilar, Datagram Routing for Internet Multicasting, *Computer Communications Review (USA) 14*,2 (1984), 58-63 .

2. M. Ahamad, M. H. Ammar, J. M. Bernabeu-Arban and M. Khalidi, Using Multicast Communication to Locate Resources in LAN-Based Distributed System, *Proceedings of the 13th Conference on Local Computer Networks*, Minneapolis, Minnesota, 1988.

3. *FDDI Token Ring Media Access Control Standard*, American National Standards Institute, Feb. 1986. Draft proposed Standard X3T9.5/83-16, Rev. 10.

4. F. Backes, Transparent Bridges for Interconnection of IEEE 802 LANs, *IEEE Network 2*,1 (January 1988).

5. J. P. Banatre, M. Banatre, G. Lapalme and F. Ployette, The Design and Building of Enchere, A Distributed Electronic Marketing System, *Communications of the ACM 29*,1 (January 1986), 19-29.

6. K. Birman and T. Joseph, Reliable Communication in the Presence of Failures, *ACM Transactions on Computer Systems 5*,1 (February 1987), 47-76.

7. A. Birrell, R. Levin, R. M. Needham and M. D. Schroeder, Grapevine: An Exercise in Distributed Computing , *Communications of the ACM 25*,4 (April 1982), 260-274.

8. J. Chang and N. F. Maxemchuk, Reliable Broadcast Protocols, *ACM Transactions on Computer Science 2*,3 (Aug. 1984), 251-273.

9. D. R. Cheriton and S. E. Deering, Host Groups: A Multicast Extension for Datagram Internetworks, *Proc. of the Ninth Data Communications Symposium*, Whistler Mountain, BC, Canada, Sep. 1985, 172-179.

10. D. R. Cheriton and W. Zwaenepoel, Distributed Process Groups in the V Kernel, *ACM Transactions on Computer Systems 3*,2 (May 1985), 77-107.

11. D. Cheriton, *VMTP: Versatile Message Transaction Protocol -- Protocol Specification*, Stanford University, February 1988. Version 0.7.

12. D. Cheriton and C. L. Williamson, VMTP as the Transport Layer for High-Performance Distributed Systems, *IEEE Communications Magazine*, June 1989, 37-44.

13. G. Chesson, The Protocol Engine Project, *Unix Review*, September 1987.

14. M. Cohn, Functional Addressing: Another Way of Looking at Multicast, *Transfer 2* ,6 (November/December 1989), 13-15.

15. D. Comer, *Internetworking with TCP/IP*, Prentice-Hall, Englewood Cliffs, New Jersey, 1988.

16. E. C. Cooper, Circus: A Replicated Procedure Call Facility, *Fourth Symposium on Reliability in Distributed Software and Database Systems*, 1984.

17. J. Crowcroft and K. Paliwoda, A Multicast Transport Protocol , *CCR 18*,4 (Aug. 1988), 247-256.

18. Y. K. Dalal and R. M. Metcalfe, Reverse Path Forwarding of Broadcast Packets, *Comm. ACM 21*,12 (Dec. 1978), 1040-1048.

19. S. E. Deering and D. Cheriton, Multicast Routing in Datagram Internetworks and Extended LANs, *Computer Communications Review 18*,4 (Aug. 1988).

20. B. J. Dempsey and A. C. Weaver, Multicast Strategies for XTP, PEI Document 90-5, January 1990.

21. *The Ethernet: A Local Area Network — Data Link Layer and Physical Layer Specifications*, Digital Equipment Corporation, Intel Corporation, Xerox Corporation, November 1982.

22. R. Dixon and D. Pitt, Addressing, Bridging, and Source Routing , *IEEE Network 2*,1 (January 1988).

23. A. J. Frank, L. D. Wittie and A. J. Bernstein, Multicast Communication on Network Computers, *IEEE Software*, May 1985, 49-61.

24. *IEEE Standard 802.1D MAC Bridges*, Institute for Electrical and Electronic Engineers Project 802— Local and Metropolitan Area Network Standards , 1985 .

25. *IEEE Standard 802.3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, Institute of Electrical and Electronics Engineers, 1985.

26. *IEEE Standard 802.4 Token-Passing Bus Access Method and Physical Layer Specifications*, Institute of Electrical and Electronics Engineers, 1985.

27. *IEEE Standard 802.5 Token Ring Access Method and Physical Layer Specifications*, Institute of Electrical and Electronics Engineers, 1985.

28. *Information Processing Systems - Open Systems Interconnection - Basic Reference Model*, International Organization for Standardization, Oct. 1984. Draft International Standard 7498.

29. M. F. Kaashoek, A. S. Tanenbaum, S. F. Hummel and H. E. Bal, An Efficient Reliable Broadcast Protocol , *Operating Systems Review 23*,4 (October 1989).

30. J. Kramer, J. Magee and A. Lister, CONIC: An Integrated Approach to Distributed Computer Control Systems, *IEE Proceedings Part E 130*,1 (January 1983), 1-10.

31. J. Lederburg and K. U. K., Towards a National Collaboratory, *Report of an Invitational NFS Workshop*, March 1989.

32. L. Liang, S. T. Chanson and G. W. Neufield, Process Groups and Group Communications: Classifications and Requirements, *IEEE Computer 23*,2 (February 1990), 56-66.

33. D. T. Marlow, Requirements for a High Performance Transport Protocol for Use on Naval Platforms, Revision 1, Naval Surface Warfare Center, July 1989.

34. P. McKinley and J. Liu, " Multicast Tree Construction in Bus-Based Computer Networks", *Communications of the ACM 33*,1 (January 1990), 29-42.

35. R. Perlman, A. Harvey and G. Varghese, Choosing the Appropriate ISO Layer for LAN Interconnection, *IEEE Network 2*,1 (January 1988), 81-85.

36. *Xpress Transfer Protocol Definition: Revision 3.4*, Protocol Engines, Incorporated, Santa Barbara, California, July 1989.

37. B. Rajagopalan and P. McKinley, A Token-Based Protocol for Reliable, Ordered Multicast Communication, *Proceedings of Eighth Symposium on Reliable Distributed Systems* , Seattle, Washington , October 1989.

38. S. Ramakrishnan and B. Jain, A Negative Acknowledgement with Periodic Polling Protocol for Multicast over LANs, *IEEE INFOCOM 1987: The Conference on Computer Communications Proceedings*, San Francisco, California, April 1987.

39. R. Sanders, *The Xpress Transfer Protocol (XTP): A Tutorial* , University of Virginia, 1989 .

40. W. D. Sincoskie and C. J. Cotton, Extended Bridge Algorithms for Large Networks, *IEEE Network* 2,1 (January 1988), 16-24.

41. W. Stallings, *Handbook of Computer Communications Standards, Volume 1: The Open Systems Interconnection (OSI) Model and OSI-Related Standards*, Macmillan, Inc., 1987.

42. *TMS 380 Adapter Chipset User's Guide, Revision D* , Texas Instruments , 1986.

43. D. W. Wall, Mechanisms for Broadcast and Selective Broadcast (excerpts from), 190, Computer Systems Laboratory, Stanford University, June 1980.

44. W. A. Wulf, *The National Collaboratory — A White Paper* , National Science Foundation , December 1988 .